Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1.   (Currently amended) A computer-implemented character validation method comprising the steps of:

receiving a character stream, wherein characters in the character stream are defined in accordance with a given markup language specification;

retrieving a data value from [[a]] the character stream; and

determining a validity of whether a character represented by said data value is a valid character as defined in the given markup language specification, wherein determining whether said character represented by said data value is a valid character comprises:

[[by]] locating a member of a data structure, said member having a direct correspondence to said retrieved data value, and

determining wherein said validity is determined whether said retrieved data value represents a valid character within the given markup language specification according to a logical combination of a plurality of status values in said located member of said data structure, wherein the determining step determines the data value's validity as a character within given computer language.

2.   (Currently amended) The computer-implemented method of claim 1 wherein said data structure comprises an array, and further comprising the step of wherein locating the member of the data structure comprises:

indexing into said array using said retrieved data value; and

pointing to said wherein a member [[of]] in said array corresponding to said data value is pointed to in response to said indexing step.

3.   (Canceled)

4.   (Currently amended) The computer-implemented method of claim 1, wherein determining whether the retrieved data value represents a valid character within the given computer language according to a logical combination of a plurality of status values in said located member of said data structure, comprises determining whether the logical combination of the plurality of status values

corresponds to a logically "TRUE" value, wherein, if the logical combination <u>of the plurality of status values</u> corresponds to a logically "TRUE" value, said data value represents a valid character.

5.   (Currently amended) The computer-implemented method of claim 1 further comprising the step of, if each character in said character stream is <u>a</u> valid <u>character</u>, applying a predetermined set of syntactic rules to byte patterns comprising said character stream.

6.   (Previously presented) The computer-implemented method of claim 1 further comprising the step of generating said data structure.

7.   (Currently amended) The computer-implemented method of claim 5 wherein said ~~character stream comprises characters in accordance with a specification for an extensible~~ <u>given</u> markup language <u>comprises an extensible markup language</u>, ~~and wherein said status values are set in accordance with a set of valid characters defined in said specification for said extensible markup language~~.

8.   (Currently amended) The computer-implemented method of claim 7 wherein the extensible markup language comprises XML ~~and wherein said syntactic rules include rules in accordance with XML~~.

9-24.   (Canceled)

25.   (Currently amended) A character validation method comprising the steps of:
          <u>receiving a character stream, wherein characters in the character stream are defined in accordance with a given extensible markup language specification;</u>
          retrieving a data value from [[a]] <u>the</u> character stream;
          determining <u>a</u> ~~validity of~~ <u>whether</u> a character represented by said data value <u>is a valid character as defined in the given extensible markup language specification, wherein determining whether said character represented by said data value is a valid character comprises:</u>
               [[by]] locating a member of a data structure, said member having a direct correspondence to said <u>retrieved</u> data value, <u>and</u>
               <u>determining whether said retrieved data value represents a valid character within the given markup language specification</u> ~~wherein said validity is determined~~ according to a logical combination of a plurality of status values in said member of said data structure, ~~wherein said character stream comprises characters in accordance with a specification for an extensible markup~~

~~language, and~~ wherein a first status value of said plurality of status values indicates whether said data value represents a valid character having a first attribute corresponding to said first status value, and <u>wherein</u> a second status value of said plurality of status values indicates whether said data value represents a valid character having a second attribute corresponding to said second status value~~, wherein the determining step determines the data value's validity as a character within a given computer language~~; and

[[if]] <u>responsive to</u> each character in said <u>character</u> stream [[is]] <u>being a</u> valid <u>character,</u> applying a predetermined set of syntactic rules to byte patterns comprising said character stream in accordance with said extensible markup language.

26.   (Currently amended) The method of claim 25 wherein said character stream comprises a message packaged in accordance with ~~a an~~ <u>the</u> extensible markup language, <u>and wherein</u> said first status value indicates whether said data value is a valid base character, said second status value indicates whether said data value is a valid digit character, and a third status value indicates whether said data value is a valid extender character.

27.   (Currently amended) The computer-implemented character validation method of claim 1 wherein <u>characters in</u> said character stream ~~comprises~~ <u>comprise</u> characters <u>defined</u> in accordance with a specification for an extensible markup language<u>,</u> and ~~within~~ <u>wherein</u> said plurality of status values[[,]] <u>comprises</u> a first status value <u>that</u> indicates whether said data value represents a valid base character, a second status value <u>that</u> indicates whether said data value represents a valid digit character, and a third status value <u>that</u> indicates whether said data value is a valid extender character.

28-29. (Canceled)